UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/902,812	07/10/2001	Andres Hejlsberg	MS1-866US	6426

22801   7590   07/09/2008
LEE & HAYES PLLC
421 W RIVERSIDE AVENUE SUITE 500
SPOKANE, WA 99201

EXAMINER
CAO, DIEM K

ART UNIT	PAPER NUMBER
2194	

MAIL DATE	DELIVERY MODE
07/09/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

	Application No.	Applicant(s)
Office Action Summary	09/902,812	HEJLSBERG ET AL.
	Examiner	Art Unit	
	DIEM K. CAO	2194	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE <u>3</u> MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
  Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on <u>25 March 2008</u>.

2a)☒ This action is **FINAL**.   2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) <u>1,3-16,18-31 and 33-42</u> is/are pending in the application.

    4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) <u>1,3-16,18-31 and 33-42</u> is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

    Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

    Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

    a)☐ All   b)☐ Some * c)☐ None of:

      1.☐ Certified copies of the priority documents have been received.

      2.☐ Certified copies of the priority documents have been received in Application No. _____.

      3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

    * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1)☒ Notice of References Cited (PTO-892)

2)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3)☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date <u>1/24/08,6/3/08</u>.

4)☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.

5)☐ Notice of Informal Patent Application

6)☐ Other: _____.

DETAILED ACTION

1.   Claims 1, 3-16, 18-31 and 33-42 are pending. Applicant has amended claims 1, 5, 16, 28,

29 and 31 and canceled claim 32.

Claim Rejections - 35 USC § 112

2.   The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making
> and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it
> pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode
> contemplated by the inventor of carrying out his invention.

3.   Claims 5-15, 16,18-27 and 42 are rejected under 35 U.S.C. 112, first paragraph, as

failing to comply with the written description requirement. The claim(s) contains subject matter

which was not described in the specification in such a way as to reasonably convey to one skilled

in the relevant art that the inventor(s), at the time the application was filed, had possession of the

claimed invention.

        Claim 5 recites "wherein the application program interface layer … and included as part

of a common language specification for local execution by an operating system or an object

model service" which is not supported by the specification. As set forth in Figs. 1 and 2 and

associated text, the application program interface, applications and the common language

specification are different entities, neither the application program interface layer nor the

application is part of the common language specification. It is unclear which one of the

application program interface layer or the application is included as part of a common language

specification. The specification seems to disclose the common language specification allows

designers of various languages to write code that is able to access underlying library

functionality, the common language specification functions as a contract between language

designers and libraries designers that can be used to promote language interoperability

(specification, page 8, lines 18-22).

Claims 6-15 fail to remedy the deficiencies of claim 5 above, and therefore are rejected

under the same ground of rejection.

Claim 16 suffer the same problem as claim 5 above, except in this claim, a common

language runtime layer is in problem instead of the application program interface in claim 5, and

therefore are rejected under the same ground of rejection.

Claims 18-27 and 42 fail to remedy the deficiencies of claim 16 above, and therefore are

rejected under the same ground of rejection.

Claim Objections

4.   Claim 1 and 3-15 are objected to because of the following informalities: claim 1 recites

"the application program interface comprising various type" on lines 14-15, "the application

program interface <u>layer</u>" should be used instead of only "the application program interface".

Appropriate correction is required. Applicant should review the entire claims carefully for any

similar problem.

5.   Claim 33 is objected to because of the following informalities: claim 33 is depended on

canceled claim 32.

Claim Rejections - 35 USC § 103

6.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in
> section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are
> such that the subject matter as a whole would have been obvious at the time the invention was made to a person
> having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the
> manner in which the invention was made.

7.   **Claims 1, 3-5, 16 and 41-42 are rejected under 35 U.S.C. 103(a) as being**

unpatentable over Wille (Presenting C#) in view of Albahari (A Comparative Overview of

C#).

        As to claim 1, Wille teaches a software architecture for a distributed computing system

comprising:

        - an application configured to handle requests (a server that can provide a web page; page

112, section 'Building the component') submitted by remote devices over a network (a client

application; page 115, section 'Creating a Simple Client Application'), and

        - a multi-tiered framework comprising (The Virtual Object System; page 21):

                - an application program interface layer (The entire C# class model is built on top

        of the NGWS runtime's Virtual Object system; page 9, second paragraph and The VOS

        type system; pages 21-23) organized into multiple root namespaces (System, System.Net,

        System.IO, System.Text; page 112, Listing 8.1 and Presenting.Csharp; page 122, Listing

        8.5 and You have already often used namespaces, such as System and System.Net. C#

        uses namespace … program; page 117, section 'Working with Namespaces'), the

        application program interface layer to present functions used by the application to access

        network and computing resources of the distributed computing system (WebRequest,

        WebResponse, retrieve a Web page from the server; page 112, section 'Building the

Component'),

     - a common language runtime layer (The NGWS runtime; page 16 and The

Virtual Execution System (VES); pages 22 and 28-29, section "Virtual Execution

System"), wherein calls to the application program interface layer are handed to the

common language runtime layer supporting applications written in different languages

(NGWS runtime environment; page 16-17, section 'NGWS Runtime') and translated into

an intermediate supported language (managed code and meta-data; pages 16-17, section

'NGWS Runtime'), the application program interface layer comprising various types

(Text, Socket, IO; page 112, Listing 8.1), wherein the types define a collection of classes

(page 48-49, section 'The class type'), interfaces (An interface declares a reference type

that has abstract member only; pages 49-50, section 'Interfaces'), delegate (A delegate …

pointers; page 50, section 'Delegates'), enumerations (Enumeration Type; page 47,

section 'Enumeration Types'), and structures (struct Type can declare … nested type;

pages 46-47, section 'struct Type') which belong to a group assigned a group name

associated with one of the root namespaces, and wherein each of the types is referenced

by a hierarchical name comprising a top level identifier prefixed to the group name (C#

uses namespaces to organize programs, and the hierarchical nature of the organization

makes it easy to present elements of a program to other programs, when you access an

element in the namespace, you either have to fully qualify it with the namespace

identifier; page 117, section 'Working with Namespaces'), and

     - an operating system layer or an object model service, wherein the calls handed

to the common language runtime layer are executed locally by the operating system layer

or the object model service layer (pages 164 -166, section "Platform Invocation Services").

Wille does not explicitly teach the application interface comprising various types related to constructing user interfaces. However, Wille teaches Console, or Presenting.Csharp which related to written to console for client application (Presenting.Csharp; page 122, Listing 8.5 and page 116, listing 8.2). Albahary teaches the application interface comprising various types related to constructing user interface (Winform library; page 17, section 15 'Libraries', and page 18, section 'Platform Interoperability' and label.font.bold=true; page 2, section Properties).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to apply the teaching of Albahary to the system of Wille to get full understanding how to construct user interface of the client application utilizing libraries supported by C# and .NET framework.

As to claim 3, Wille teaches wherein the distributed computing system comprises client devices and server devices that handle requests from the client devices, the remote devices comprising at least one client device (a server that can provide a web page; page 112, section 'Building the component', a client application; page 115, section 'Creating a Simple Client Application').

As to claim 4, Wille teaches the distributed computing system comprises client devices and server devices that handle requests from the client devices, the remote devices comprising at least one server device that is configured as a Web server (a server that can provide a web page;

page 112, section 'Building the component', a client application; page 115, section 'Creating a

Simple Client Application' and WebRequest, WebResponse, WebRequestFactory; page 113,

lines 31-33).

As to claim 41, Wille teaches the various types comprises classes (page 48-49, section

'The class type'), interfaces (An interface declares a reference type that has abstract member

only; pages 49-50, section 'Interfaces'), delegates (A delegate … pointers; page 50, section

'Delegates'), structures (struct Type can declare … nested type; pages 46-47, section 'struct

Type') and enumerations (Enumeration Type; page 47, section 'Enumeration Types').

As to claim 5, Wille teaches A multi-tiered architecture (The Virtual Object System; page

21) including an application program interface layer ((The entire C# class model is built on top

of the NGWS runtime's Virtual Object system; page 9, second paragraph and The VOS type

system; pages 21-23) comprising multiple types (System, System.Net, System.IO, System.Text;

page 112, Listing 8.1 and Presenting.Csharp; page 122, Listing 8.5 and You have already often

used namespaces, such as System and System.Net. C# uses namespace … program; page 117,

section 'Working with Namespaces'), the individual type being associated with one or more

group and being references by one or more hierarchical names, wherein each hierarchical name

includes a top level identifier prefixed to a group name assigned to one of the one or more groups

(C# uses namespaces to organize programs, and the hierarchical nature of the organization makes

it easy to present elements of a program to other programs, when you access an element in the

namespace, you either have to fully qualify it with the namespace identifier; page 117, section

'Working with Namespaces'), the type comprising classes which represent managed heap

allocated data that has reference assignment semantics (page 48-49, section 'The class type'),

interfaces that define a contract that other types can implement (An interface declares a reference

type that has abstract member only; pages 49-50, section 'Interfaces'), delegates that are object

oriented function pointers (A delegate … pointers; page 50, section 'Delegates'), structures that

represent static allocated data that has value assignment semantic (struct Type can declare …

nested type; pages 46-47, section 'struct Type'), and enumerations which are value types that

represent named constants (Enumeration Type; page 47, section 'Enumeration Types'), wherein

the application program interface is associated with a common language runtime layer

supporting applications written in one or more of several different languages (NGWS runtime

environment; page 16-17, section 'NGWS Runtime') an translated into an intermediate language

supported by the common runtime layer (managed code and meta-data; pages 16-17, section

'NGWS Runtime') and including as part of a common language specification (pages 24-27,

section "The Common Language Specification") for local execution by an operating system or

an object model service (pages 164 -166, section "Platform Invocation Services").

      Wille does not explicitly teach the application interface comprising various types related

to constructing user interfaces. However, Wille teaches Console, or Presenting.Csharp which

related to written to console for client application (Presenting.Csharp; page 122, Listing 8.5 and

page 116, listing 8.2). Albahari teaches the application interface comprising various types

related to constructing user interface (Windorm library; page 17, section 15 'Libraries', and page

18, section 'Platform Interoperability' and label.font.bold=true; page 2, section Properties).

      It would have been obvious to one of ordinary skill in the art at the time the invention

was made to apply the teaching of Albahary to the system of Wille to get full understanding how

to construct user interface of the client application utilizing libraries supported by C# and .NET

framework.

As to claim 16, Wille teaches

- one or more applications configured to be executed on one or more computing devices,

    the applications handling requests submitted from remote computing devices (a server

    that can provide a web page; page 112, section 'Building the component', a client

    application; page 115, section 'Creating a Simple Client Application'),

- a networking platform to support the one or more applications (NGWS framework; page

    112, section 'Building the Component'),

- an application programming interface to interface the one or more applications with the

    networking platform (System, System.Net, System.IO, System.Text; page 112, Listing

    8.1 and Presenting.Csharp; page 122, Listing 8.5), the application programming interface

    comprising various types (Net, IO, Text; page 112, Listing 8.1), individual types being

    associated with one or more groups and being referenced by one or more hierarchical

    names, wherein each of the hierarchical names includes a top level identifier prefixed to

    group name assigned to one of the one or more groups (C# uses namespaces to organize

    programs, and the hierarchical nature of the organization makes it easy to present

    elements of a program to other programs, when you access an element in the namespace,

    you either have to fully qualify it with the namespace identifier; page 117, section

    'Working with Namespaces'), and

- a common language runtime layer supported applications written in one or more different
   languages and translated into an intermediate language supported by the common runtime
   layer (NGWS runtime environment; page 16-17, section 'NGWS Runtime', managed
   code and meta-data; pages 16-17, section 'NGWS Runtime') and including as part of a
   common language specification (pages 24-27, section "The Common Language
   Specification") for local execution by an operating system or an object model service
   (pages 164 -166, section "Platform Invocation Services").

Wille does not explicitly teach the application interface comprising various types related
to constructing user interfaces. However, Wille teaches Console, or Presenting.Csharp which
related to written to console for client application (Presenting.Csharp; page 122, Listing 8.5 and
page 116, listing 8.2). Albahary teaches the application interface comprising various types
related to constructing user interface (Windorm library; page 17, section 15 'Libraries', and page
18, section 'Platform Interoperability' and label.font.bold=true; page 2, section Properties).

It would have been obvious to one of ordinary skill in the art at the time the invention
was made to apply the teaching of Albahary to the system of Wille to get full understanding how
to construct user interface of the client application utilizing libraries supported by C# and .NET
framework.

As to claim 42, see rejection of claim 41 above.

8.   **Claims 6-15, 18-31 and 33-40 are rejected under 35 U.S.C. 103(a) as being**

unpatentable over Wille (Presenting C#) in view of Albahari (A Comparative Overview of

C#) **further in view of Williams (Teach Yourself Visual C++ 6 in 24 Hours).**

As to claim 6, Wille and Albahari do not explicitly teach the classes comprise a form

class that represents a window or a dialog box that makes up an application's user interface.

However, Wille teaches C# is derived from C and C++ (page 6, second and third paragraphs).

Williams teaches the classes comprise a form class that represents a window or a dialog box that

makes up an application's user interface (A form view is a view that can contain controls, much

like a dialog box; pages 369-370). It would have been obvious to one of ordinary skill in the art

at the time the invention was made to apply the teaching of Williams to the system of Wille and

Albahari because Williams teaches how to using classes of C++ when developing an application

user interface.

As to claim 7, Williams teaches the form class has multiple members comprising one or

more of public static properties, public static methods, public instance constructors, public

instance methods, public instance properties, public instance events, protected instance

properties, and protected instance methods (see section "Handling Events and Messages in the

Form View Class", pages 375-376).

As to claim 8, Williams teaches the interfaces comprise a button control interface that

allows a control to act like a button on a form (Button Controls; see pages 73-74).

As to claim 9, Williams teaches the interfaces comprise a container control interface that

provides functionality for a control to act as a parent for other controls (A form view is a view that can contain controls; page 369).

As to claim 10, Williams teaches the interfaces comprise an editing notification interface (Edit Controls; pages 89-91).

As to claim 11, Williams teaches interfaces comprise a data object interface that provides a format independent mechanism for transferring data (Serialization, the MFC Classes Used for Serialization; page 352-353).

As to claim 12, Williams teaches the interfaces comprise a feature support interface that specifies a standard interface for retrieving feature information from a current system (CFile; see Microsoft Foundation Class).

As to claim 13, Williams teaches the interfaces comprise a message filter interface (Handling Messages with ClassWizard; pages 127-128).

As to claim 14, Williams teaches the interfaces comprise a handle-exposing interface to expose handles (Array; page 148).

As to claim 15, see rejections of claims 8-14 above.

As to claims 18-26, see rejections of claims 6-14bove.

As to claim 27, see rejections of claims 8-14 above.

As to claim 28, see rejection of claims 5 and 27 above. Wille further teaches a computer system including one or more microprocessors and one or more software programs (inherent from a server that can provide a web page; page 112, section 'Building the component', a client application; page 115, section 'Creating a Simple Client Application' and WebRequest, WebResponse, WebRequestFactory; page 113, lines 31-33). Albahary teaches the one or more software programs utilizing an application program interface to request services from an operating system (some of the .NET libraries are based on Windows, particularly … Windows API; page 18, section 'Platform Interoprability').

As to claim 29, Wille teaches a method comprising:

- managing network and computing resources for a distributed computing system (web page from a server; page 112, section "Building the Component"),

- exposing a set of functions that enable developers to access the network and computing resources of the distributed computing system (WebRequest, WebResponse, retrieve a Web page from the server; page 112, section 'Building the Component'), the set of functions comprising functions that are grouped into multiple namespaces (System, System.Net, System.IO, System.Text; page 112, Listing 8.1 and Presenting.Csharp; page 122, Listing 8.5 and You have already often used namespaces, such as System and System.Net. C# uses namespace … program;

page 117, section 'Working with Namespaces') that define a collection of classes which

represent managed heap allocated data that has reference assignment semantics (page 48-49,

section 'The class type'), interfaces that define a contract that other types can implement (An

interface declares a reference type that has abstract member only; pages 49-50, section

'Interfaces'), delegates that are object oriented function pointers (A delegate … pointers; page

50, section 'Delegates'), enumerations which are value types that represent named constants

(Enumeration Type; page 47, section 'Enumeration Types') and structures that represent static

allocated data that has value assignment semantics (struct Type can declare … nested type;

pages 46-47, section 'struct Type'); and

     - using a common language runtime layer supporting applications written in one or more

different languages and translated into an intermediate language supported by the common

runtime layer (The NGWS runtime; page 16 and The Virtual Execution System (VES); pages 22

and 28-29, section "Virtual Execution System", NGWS runtime environment; page 16-17,

section 'NGWS Runtime', managed code and meta-data; pages 16-17, section 'NGWS

Runtime').

     Wille does not explicitly teach the set of functions comprising functions to facilitate

construction of user interface, wherein the user interfaces include windowing, menus, and

dialogs. However, Wille teaches Console, or Presenting.Csharp which related to written to

console for client application (Presenting.Csharp; page 122, Listing 8.5 and page 116, listing

8.2). Albahary teaches the application interface comprising various types related to constructing

user interface (Windorm library; page 17, section 15 'Libraries', and page 18, section 'Platform

Interoperability' and label.font.bold=true; page 2, section Properties). Williams teaches the user

interface include windowing (CWindowDC; Microsft Foundation Class Hierarchy – Version

6.0), menus (CMenu; Microsft Foundation Class Hierarchy – Version 6.0), and dialogs (dialog

boxes; see pages 55-56).

It would have been obvious to one of ordinary skill in the art at the time the invention

was made to apply the teaching of Albahary to the system of Wille to get full understanding how

to construct user interface of the client application utilizing libraries supported by C# and .NET

framework. Also see claim 6 above for reason to apply the teaching of Williams to the system of

Wille and Albahary.

As to claim 30, Wille teaches receiving a request from a remote computing device, the

request containing a call to the set of functions (WebResponse, WebRequestFactory; page 113,

lines 31-33).

As to claim 31, Wille teaches creating a name space with functions (C# uses namespaces

to organize programs, and the hierarchical nature of the organization makes it easy to present

elements of a program to other programs, when you access an element in the namespace, you

either have to fully qualify it with the namespace identifier; pages 117-118, section 'Working

with Namespace' and 'Wrapping a Class in a Namespace), the namespace defining classes which

represent managed heap allocated data that has reference assignment semantics (page 48-49,

section 'The class type'), interfaces that define a contract that other types can implement (An

interface declares a reference type that has abstract member only; pages 49-50, section

'Interfaces'), delegates that are object oriented function pointers (A delegate … pointers; page

50, section 'Delegates'), structures that represent static allocated data that has value assignment

semantic (struct Type can declare … nested type; pages 46-47, section 'struct Type'), and

enumerations which are value types that represent named constants (Enumeration Type; page 47,

section 'Enumeration Types'),

Wille does not explicitly teach the set of functions comprising functions to facilitate

construction of user interface, wherein the user interfaces include windowing, menus, and

dialogs. However, Wille teaches Console, or Presenting.Csharp which related to written to

console for client application (Presenting.Csharp; page 122, Listing 8.5 and page 116, listing

8.2). Albahary teaches the application interface comprising various types related to constructing

user interface (Windorm library; page 17, section 15 'Libraries', and page 18, section 'Platform

Interoperability' and label.font.bold=true; page 2, section Properties). Williams teaches the user

interface include windowing (CWindowDC; Microsft Foundation Class Hierarchy – Version

6.0), menus (CMenu; Microsft Foundation Class Hierarchy – Version 6.0), and dialogs (dialog

boxes; see pages 55-56).

It would have been obvious to one of ordinary skill in the art at the time the invention

was made to apply the teaching of Albahary to the system of Wille to get full understanding how

to construct user interface of the client application utilizing libraries supported by C# and .NET

framework. Also see claim 6 above for reason to apply the teaching of Williams to the system of

Wille and Albahary.

As to claim 33-40, see rejection of claims 7-14 above.

Response to Arguments

9.   Applicant's arguments with respect to claims 1, 3-16, 18-31 and 33-42 have been

considered but are moot in view of the new ground(s) of rejection.

Conclusion

10.   Applicant's amendment necessitated the new ground(s) of rejection presented in this

Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a).

Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

        A shortened statutory period for reply to this final action is set to expire THREE

MONTHS from the mailing date of this action. In the event a first reply is filed within TWO

MONTHS of the mailing date of this final action and the advisory action is not mailed until after

the end of the THREE-MONTH shortened statutory period, then the shortened statutory period

will expire on the date the advisory action is mailed, and any extension fee pursuant to 37

CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event,

however, will the statutory period for reply expire later than SIX MONTHS from the date of this

final action.

        Any inquiry concerning this communication or earlier communications from the

examiner should be directed to DIEM K. CAO whose telephone number is (571)272-3760. The

examiner can normally be reached on Monday - Friday, 7:30AM - 3:00PM.

        If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Meng-Ai An can be reached on (571) 272-3756. The fax phone number for the

organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent

Application Information Retrieval (PAIR) system. Status information for published applications

may be obtained from either Private PAIR or Public PAIR. Status information for unpublished

applications is available through Private PAIR only. For more information about the PAIR

system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR

system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would

like assistance from a USPTO Customer Service Representative or access to the automated

information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

DC
July 2, 2008

/Li B. Zhen/
Primary Examiner, Art Unit 2194